آموزش یک مدل برای پیشبینی از دست رفتن مشتری شرکت ارائه دهنده اینترنت

Telco Customer Churn

علی خضری پور

Alikhezri1252@gmail.com

1.LOAD DATA

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import f1_score
from sklearn.metrics import roc_auc_score
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_selection import RFE
from sklearn.feature_selection import mutual_info_classif
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
from sklearn.linear_model import LogisticRegressionCV
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import accuracy_score
import xgboost as xgb
import lightgbm as lgb
import joblib
from datetime import datetime
```

-اولین قدم فراخوانی کتابخانه های مورد استفادمون هست

```
df=pd.read_csv(r'D:\Coding\MLdata\WA_Fn-UseC_-Telco-Customer-Churn.csv')
df
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	S
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	No	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	No	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	No	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	Yes	Yes	

7043 rows × 21 columns

-باز کردن دیتاست و برسی اولیه دیتاست

```
df.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   gender            7043 non-null   object
 1   SeniorCitizen     7043 non-null   int64
 2   Partner           7043 non-null   object
 3   Dependents        7043 non-null   object
 4   tenure            7043 non-null   int64
 5   PhoneService      7043 non-null   int64
 6   MultipleLines     7043 non-null   object
 7   InternetService   7043 non-null   object
 8   OnlineSecurity    7043 non-null   object
 9   OnlineBackup      7043 non-null   object
 10  DeviceProtection  7043 non-null   object
 11  TechSupport       7043 non-null   object
 12  StreamingTV       7043 non-null   object
 13  StreamingMovies   7043 non-null   object
 14  Contract          7043 non-null   object
 15  PaperlessBilling  7043 non-null   object
 16  PaymentMethod     7043 non-null   object
 17  MonthlyCharges    7043 non-null   float64
 18  TotalCharges      7032 non-null   float64
 19  Churn             7043 non-null   int64
dtypes: float64(2), int64(4), object(14)
memory usage: 1.1+ MB
```

-برسی نوع داده در هر ستون دیتاست و وجود null

2.Pre processing

```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'].fillna(df['tenure'] * df['MonthlyCharges'])
# encode target
df['Churn'] = df['Churn'].map({'Yes': 1, 'No': 0})
df['PhoneService'] = df['PhoneService'].map({'Yes': 1, 'No': 0})
df=df.drop('customerID',axis=1)
```

-نوع داده در ستون total charges از نوع عدد نیست پس اون رو تبدیل به عدد میکنیم

همچنین بعضی جاهای خالی رو باضرب تعداد ماه قرار داد و هزینه ماهانه به دست می آوریم

ستونchurn که ستون هدف ما هست نوع string هست و با استفاده از map اون رو به باینری میکنیم

ستون customer idچون اطلاعات خوبی به ما نمیده حذف میکنیم

```
# encode categoricals
df_encoded = df.copy()
cat_cols = df.select_dtypes(include='object')
for col in cat_cols:
    df_encoded[col] = LabelEncoder().fit_transform(df_encoded[col])
```

-لیبل گذاری برای ستون ها غیر عددی برای کار کردن با مدل ها

```
X = df_encoded.drop('Churn', axis=1)
Y = df_encoded['Churn']
```

-مشخص کردن x,y

```
impute = SimpleImputer()
impute.fit(X[['TotalCharges']])
X['TotalCharges'] = impute.transform(X[['TotalCharges']])
X = X.dropna()
X
```

-باقی مانده جاهای خالی ستون total charges رو با میانگین پر میکنیم

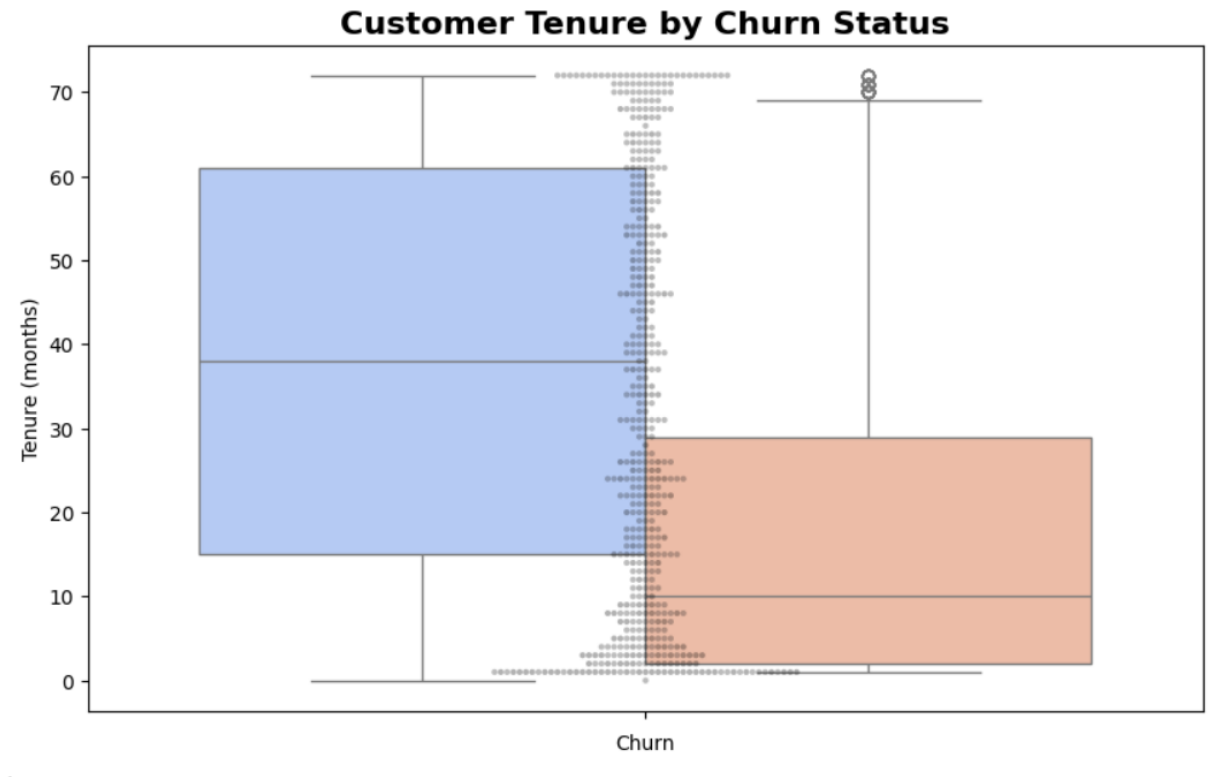
نتیجه:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport
0	0	0	1	0	1	0	1	0	0	2	0	0
1	1	0	0	0	34	1	0	0	2	0	2	0
2	1	0	0	0	2	1	0	0	2	2	0	0
3	1	0	0	0	45	0	1	0	2	0	2	2
4	0	0	0	0	2	1	0	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...
7038	1	0	1	1	24	1	2	0	2	0	2	2
7039	0	0	1	1	72	1	2	1	0	2	2	0
7040	0	0	1	1	11	0	1	0	2	0	0	0
7041	1	1	1	0	4	1	2	1	0	0	0	0
7042	1	0	0	0	66	1	0	1	2	0	2	2

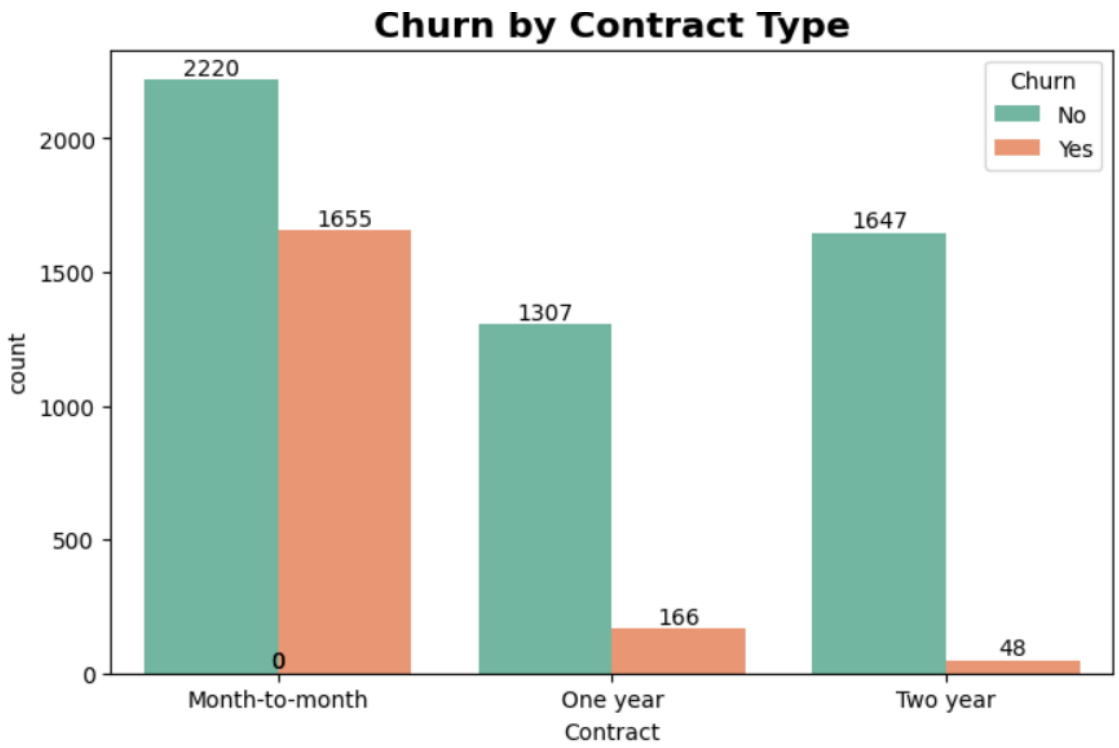
7043 rows × 19 columns

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection
count	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000
mean	0.504756	0.162147	0.483033	0.299588	32.371149	0.903166	0.940508	0.872923	0.790004	0.906432	0.904444
std	0.500013	0.368612	0.499748	0.458110	24.559481	0.295752	0.948554	0.737796	0.859848	0.880162	0.879949
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	9.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	1.000000	0.000000	0.000000	0.000000	29.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
75%	1.000000	0.000000	1.000000	1.000000	55.000000	1.000000	2.000000	1.000000	2.000000	2.000000	2.000000
max	1.000000	1.000000	1.000000	1.000000	72.000000	1.000000	2.000000	2.000000	2.000000	2.000000	2.000000

3.visualization



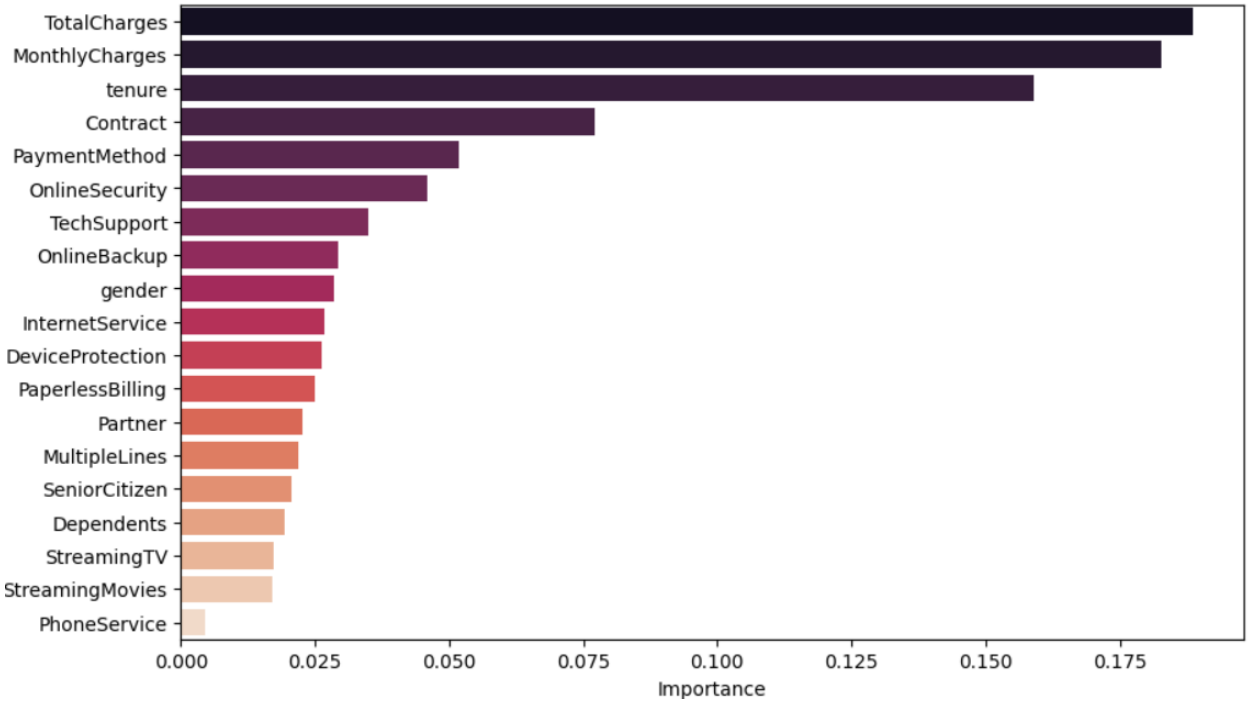
-نمودار ارتباط تعداد ماه ها با از دست رفتن مشتری



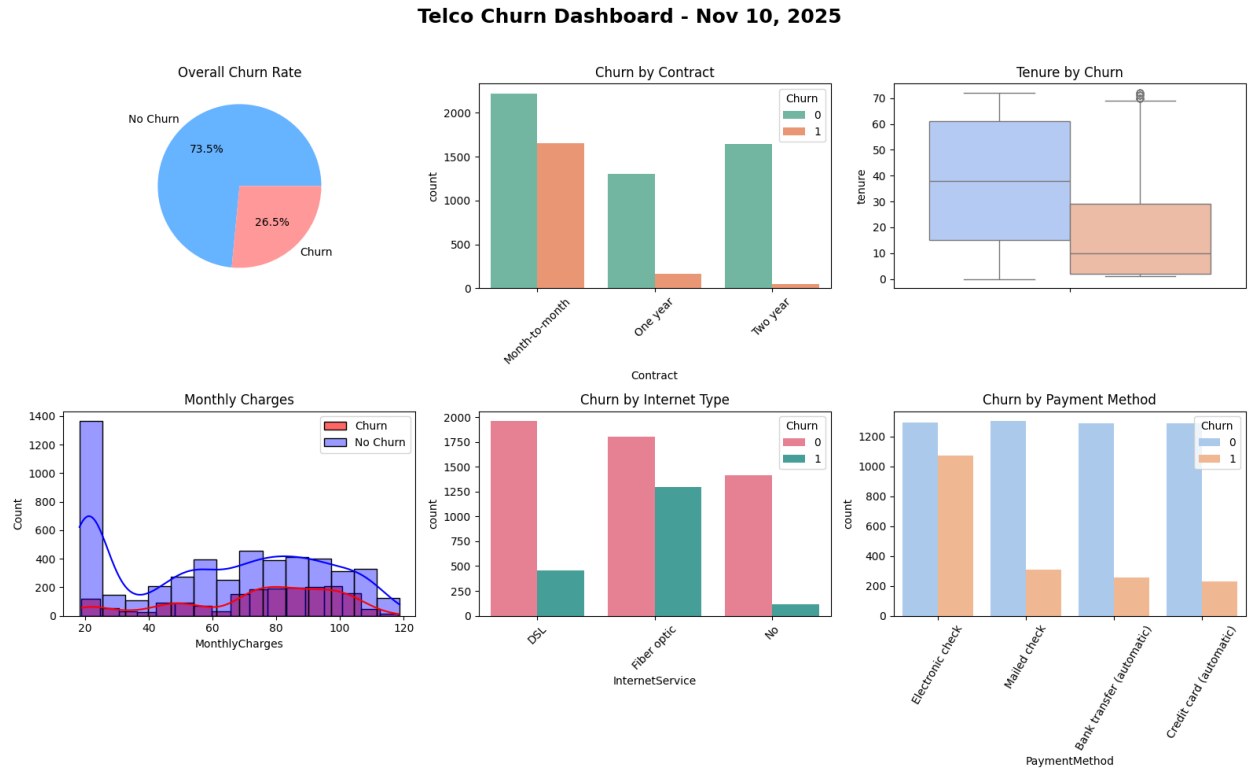
-ازدست رفتن با نوع قرار داد

```
model = RandomForestClassifier(n_estimators=100, random_state=12)
model.fit(X, Y)
feat_imp = pd.Series(model.feature_importances_, index=X.columns).sort_values(ascending=False)
plt.figure(figsize=(10,6))
sns.barplot(x=feat_imp.values, y=feat_imp.index, palette='rocket')
plt.title('Feature Importance (Random Forest)', fontsize=16, fontweight='bold')
plt.xlabel('Importance')
plt.show()
```

-استفاده از random forest برای بدست اوردن اهمیت فیچر ها



-میتونیم توی این نمودار تاثیر گذاری هر یک از فیچر ها رو ببینیم چون سه تا فیچر آخر نمودار اهمیت زیادی ندارند آنها را حذف میکنیم



نمای کلی

4.Train and evaluation

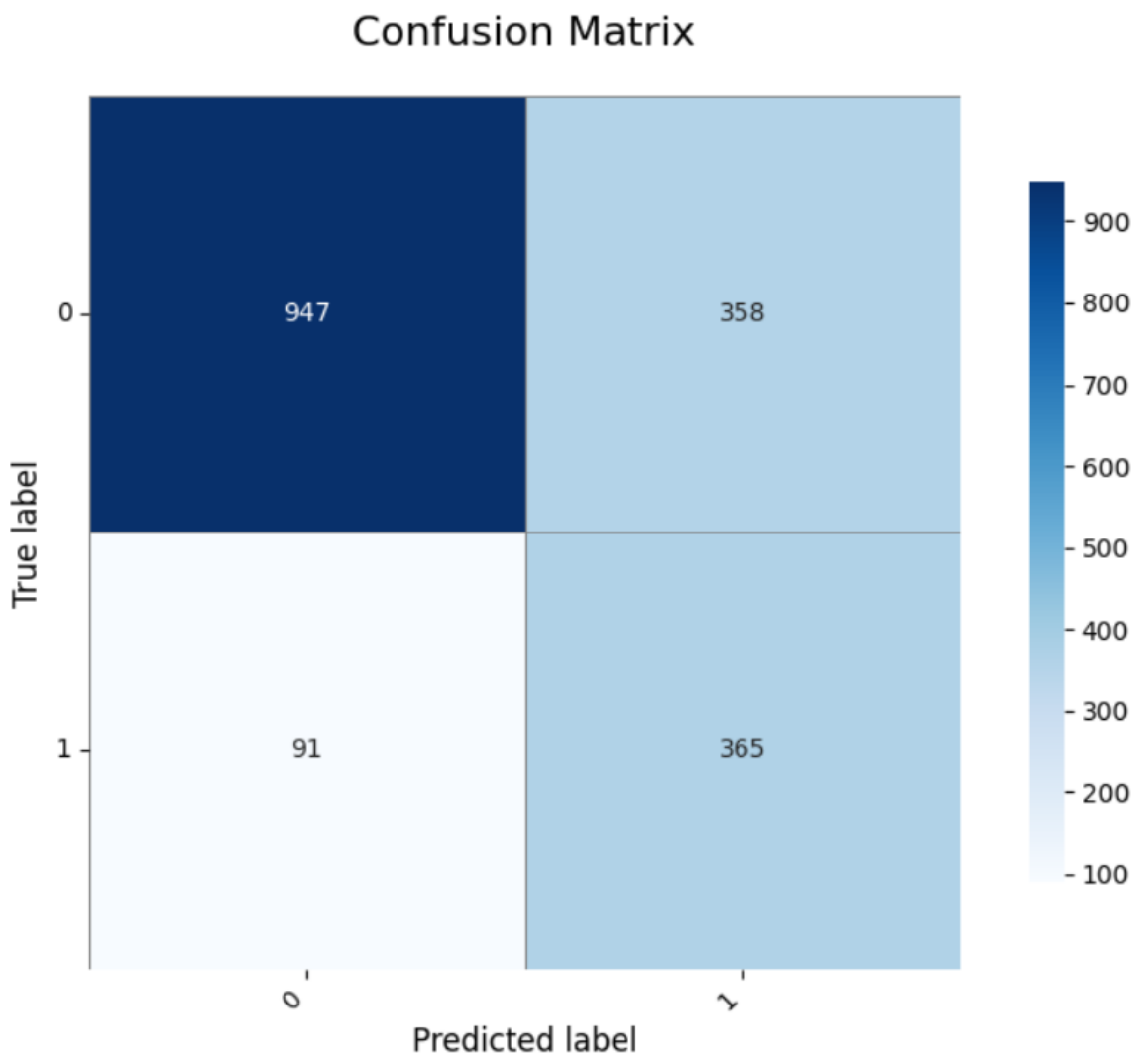
من سه مدل رو روی این دیتاست آموزش دادم ,xgboost , lightgbm
logisticregressionCV

با استفاده از grid search cv

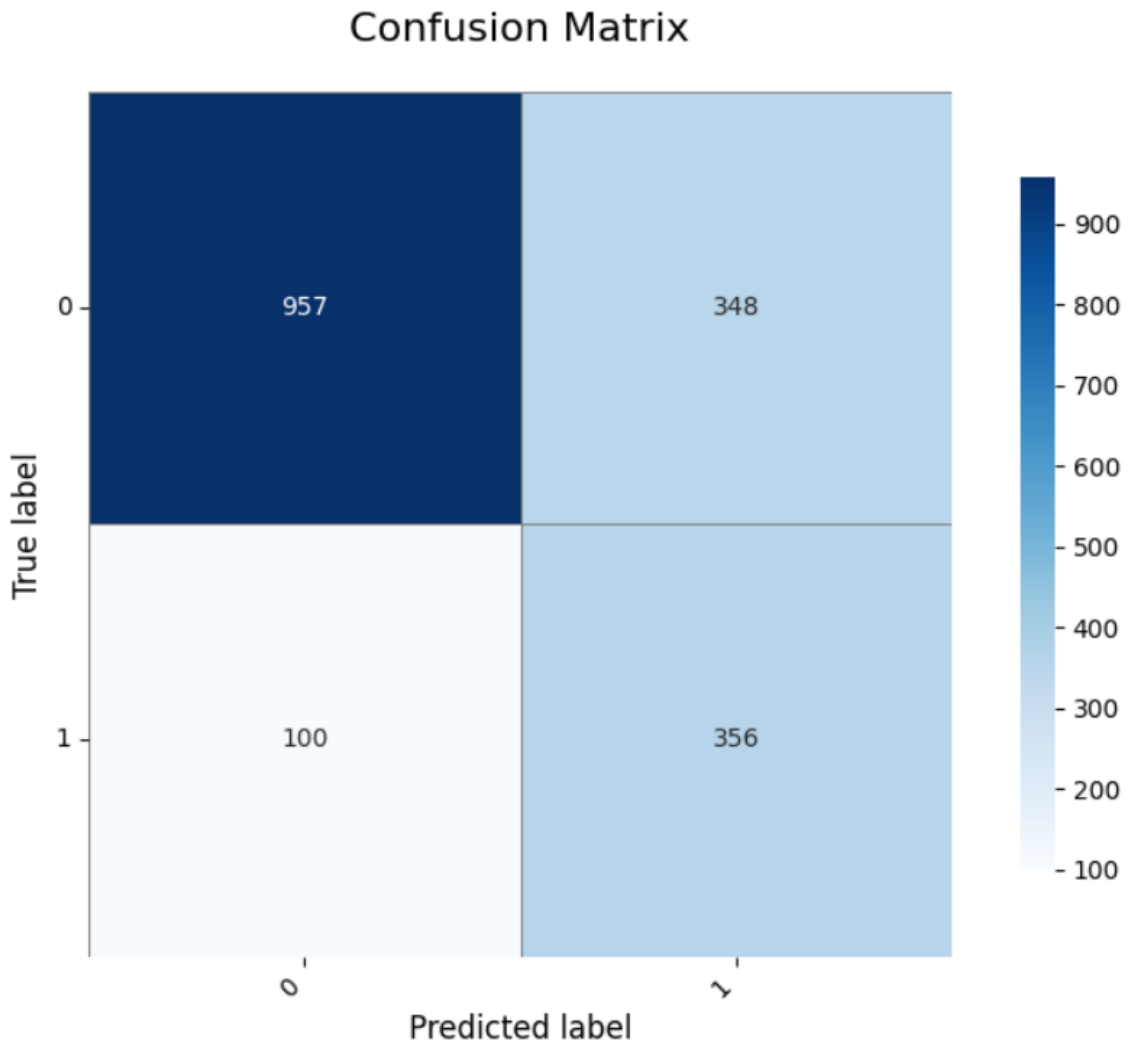
هایپر پارامتر های مناسب رو پیدا کردم و از overfitجلوگیری کردم

بعد confusion matrix رو برای هر مدل رسم کردم

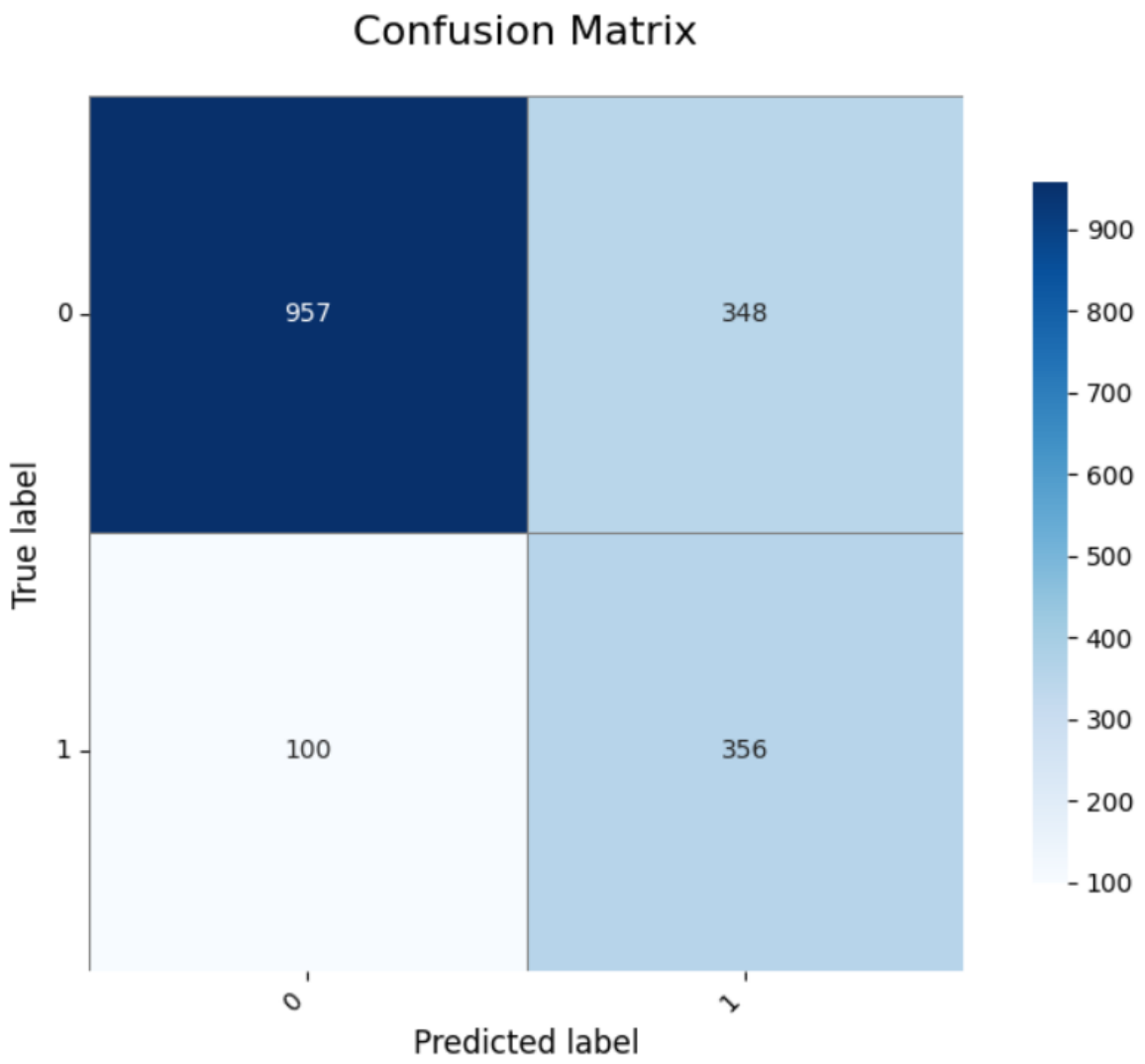
```
X_train , X_test ,Y_train , Y_test=train_test_split(X,Y)
scale_pos_weight = (Y_train == 0).sum() / (Y_train == 1).sum()
model1=xgb.XGBClassifier(objective='binary:logistic',
    eval_metric='auc',
    random_state=12,
    scale_pos_weight=scale_pos_weight,
    verbosity=0
    )

model2=lgb.LGBMClassifier(
    objective='binary',
    metric='auc',
    boosting_type='gbdt',
    class_weight='balanced',
    random_state=12,
)

model3=LogisticRegressionCV(
    Cs=20, cv=5, scoring='roc_auc',
    penalty='l2', solver='newton-cholesky',
    max_iter=10_000, n_jobs=-1, random_state=12
)
param_grid1 = {
    'max_depth': [4, 6, 8],
    'learning_rate': [0.01, 0.1],
    'n_estimators': [200, 300],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}
grid = GridSearchCV(model1, param_grid1, cv=5, scoring='roc_auc', n_jobs=-1, verbose=1)
grid.fit(X_train, Y_train)
```


Confusion Matrix
947
358
91
365
0
1
True label
0
1
Predicted label
900
800
700
600
500
400
300
200
100

```python
param_grid2 = {
    'num_leaves': [31, 127],
    'max_depth': [6, 10],
    'learning_rate': [0.01, 0.05],
    'n_estimators': [200, 300],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}
grid2 = GridSearchCV(model2, param_grid2, cv=5, scoring='roc_auc',n_jobs=-1, verbose=1)
grid2.fit(X_train, Y_train)
```



```
model3.fit(X_train,Y_train)
y_pred3=model3.predict(X_test)
cm3 = confusion_matrix(Y_test, y_pred2)
plt.figure(figsize=(8, 6))
sns.heatmap(
    cm3,
    annot=True,
    fmt="d",
    cmap="Blues",
    linewidths=0.5,
    linecolor="gray",
    cbar_kws={"shrink": 0.8},
    square=True
)
plt.title("Confusion Matrix", fontsize=16, pad=20)
plt.xlabel("Predicted label", fontsize=12)
plt.ylabel("True label", fontsize=12)
plt.xticks(rotation=45, ha="right")
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```



درنهایت با استفاده از f1 score, accuracy, recall

بهترین مدل رو پیدا کردم

نمرات:xgboost

```
score=accuracy_score(Y_test,y_pred)
score
```

✓ 0.0s

```
0.7450312322544009
```

```
FSCORE1=f1_score(Y_test,y_pred)
FSCORE1
```
✓ 0.0s

```
0.6191687871077184
```

```
RSCORE1=recall_score(Y_test,y_pred)
RSCORE1
```
✓ 0.0s

```
0.8004385964912281
```

Lightgbm

```
FSCORE2=f1_score(Y_test,y_pred2)
FSCORE2
```
✓ 0.0s

```
0.6137931034482759
```

```
score2=accuracy_score(Y_test,y_pred2)
score2
```
✓ 0.0s

```
0.7455990914253265
```

```
RSCORE2=recall_score(Y_test,y_pred2)
RSCORE2
```
✓ 0.0s

```
0.7807017543859649
```

Logisticregression

```
RSCORE3=recall_score(Y_test,y_pred3)
RSCORE3
```
✓ 0.0s

```
0.5350877192982456
```

```
FSCORE3=f1_score(Y_test,y_pred3)
FSCORE3
```
✓ 0.0s

```
0.5802615933412604
```

```
score3=accuracy_score(Y_test,y_pred3)
score3
```
✓ 0.0s

```
0.7995457126632595
```

بهترین مدل از نظر من xgboost بود به خاطر نمره recall بالاتر در اینجا **true positive برای ما مهم تره**

درنهایت ذخیره مدل

```
# save best model
model_path = f'xgb_churn_model_{datetime.now().strftime("%Y%m%d_%H%M")}.pkl'
joblib.dump(model1, model_path)
print(f"model path: {model_path}")

# save predictions
preds = pd.DataFrame({
    'y_true': Y_test.values,
    'y_pred': y_pred,
})
preds.to_csv('xgb_predictions.csv', index=False)
```

```
odel path: xgb_churn_model_20251111_1130.pkl
```

https://github.com/aqua1252/my-projects/blob/main/xgb_predictions.csv

https://github.com/aqua1252/my-projects/blob/main/xgb_churn_model_20251111_1130.pkl

فایل کد

https://github.com/aqua1252/my-projects/blob/main/Telco%20Customer%20Churn.ipynb